How to install a Debian 12 server with Apache, PostgreSQL and PHP ?

RASTELLI Maxime

Introduction

You may want to install a server like this if you want to host your website made with PHP or your PostgreSQL server with a graphical interface available on a website, this is why this handbook will be useful. In this handbook, I will do everything on a virtual machine with Qemu but except for some steps, it will work on real hardware.

I. Install Debian 12 on your server

1. Requirements

You need at least a Debian 12 (Bookworm) disk image file with the tag “netinst” and for processor x86 64 bits. You can check if you have the good one with the fingerprint with the command sha512sum iso_file, you should get this fingerprint:

33c08e56c83d13007e4a5511b9bf2c4926c4aa12fd5dd56d493c0653aecbab380988c5bf1671dbaea75c582827797d98c4a611f7fb2b131fbde2c677d5258ec9

Because it’s the “netinst” version, you need an internet connection to make the installation, if you don’t have one, you can download a DVD version which doesn’t require any internet connection.

2. Launching the virtual machine

To launch the Virtual Machine, we will put this command in a shell:

qemu-system-x86_64 -machine q35 -cpu host -m 4G -enable-kvm -device VGA,xres=1024,yres=768 -display gtk,zoom-to-fit=off -drive $drive -device e1000,netdev=net0 -netdev user,id=net0,hostfwd=tcp::2222-:22,hostfwd=tcp::4443-:443,hostfwd=tcp::8080-:80,hostfwd=tcp::5432-:5432

Here’s the detail of the arguments:

qemu-system-x86_64 : call the program Qemu

-machine q35 : select the type of machine, which in this case is Q35

-cpu host : select the CPU that the VM will use, here it's the hosted CPU, the one that runs our real machine

-m 4G allow: 4Gb of RAM to the VM

-enable-kvm : enables KVM hypervisor

-device VGA,xres=1024,yres=768 : define the color mode VGA, and the resolution of the window

-display gtk,zoom-to-fit=off : select what type of display to use and disable the function to zoom in to fill the window

-drive $drive : Define which drive to use, in this case it's a variable that has the path where the drive file is located

-device e1000,netdev=net0 : define the Wifi device and the name of the interface

-netdev user,id=net0,hostfwd=tcp::2222-:22,hostfwd=tcp::4443-:443,hostfwd=tcp::8080-:80,hostfwd=tcp::5432-:5432 : defines ports redirection, for example, everything that goes throw the port 22 of the VM, will go on the port 2222 of the real machine

So the ports that has been redirected are:

	Default port	Changed port
SSH	22	2222
HTTP	80	8080
HTTPS	443	4443
PostgreSQL	5432	5432

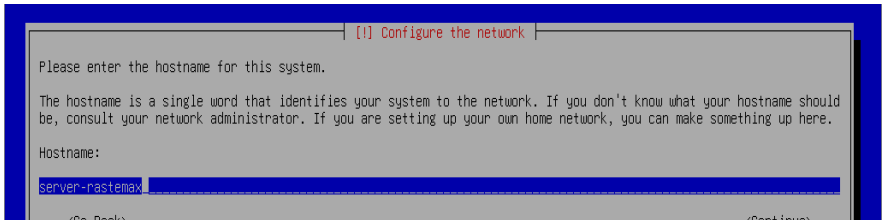
3. Install Debian 12

After launching this command, you will be received with this screen :

In the handbook, I will choose the “install” option, you can also select the “graphical install”, it doesn’t change the installation process.

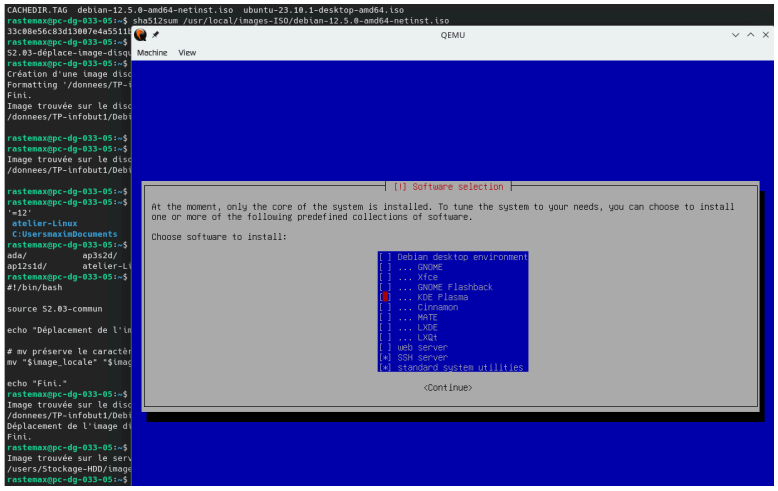
Here’s a little list of settings that I recommend you:

- Language : English
- **Location :** It depends on your location(France,England,United States,...)
- Locales : United States, en_US.UTF-8
- **Keyboard :** It depends which type of keyboard you’re using(Qwerty, Azerty,...)
- **Hostname** : for this I will named it “server-” with my



university nickname but you can put anything you want

- **Root Password** : Try to make something difficult to find, for me it will simply be “root”.
- User Account - **Full Name** : For example “Jean Toto”
- **User Name** : You can choose as you want and I will be using “rastemax”
-
- **User Password** : An easy password to remember for me is ‘etu’ but you can choose yourself
- Partition disks : Guided - use entire disk
- Partition disks : All files in one partition
- **Partition disks : Yes**
- **Software Selection** : By default, there will be “Debian Desktop” checked and some other things, make sure to uncheck them and check “ssh-server” option



- Install GRUB : Yes
- **Device for boot loader : `/dev/sda`**

After you’re done with that, the system will just restart.

4. Checking if everything is OK

To make sure the installation was successful, you can check with the command **`cat /etc/fstab`**, this file is a list of file system automatically mounted on startup and it should be like that:

```
rm: remove regular file '
rastemax@pc-dg-033-05:~$
Création d'une image disq
Formatting '/donnees/TP-i
Fini.
Image trouvée sur le disq
/donnees/TP-infobut1/Debi
rastemax@pc-dg-033-05:~$
Image trouvée sur le disq
/donnees/TP-infobut1/Debi
Déplacement de l'image di
Fini.
rastemax@pc-dg-033-05:~$
Image trouvée sur le serv
/users/Stockage-HDD/image
rastemax@pc-dg-033-05:~$
```

```
rastemax@server-rastemax:~$ cat /etc/fstab
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# systemd generates mount units based on this file, see systemd.mount(5).
# Please run 'systemctl daemon-reload' after making changes here.
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/sda1 during installation
UUID=a3ef6f9e-3bf5-4ac1-b82a-c0a436845f88 /               ext4    errors=remount-ro 0       1
# swap was on /dev/sda5 during installation
UUID=f39349d4-bc78-401d-beff-d02408a88140 none            swap    sw              0       0
/dev/sr0        /media/cdrom0   udf,iso9660 user,noauto     0       0
rastemax@server-rastemax:~$
```

We can see that we have 3 file systems.

5. Using ssh

For the rest of the handbook, we will use the root account, to access it you need to type **su -** and enter the root password you have defined.

By typing **systemctl status ssh**, you can see if the ssh is working.

```
rastemax@pc-dg-033-11:~$ S2.03-lance-n
[1] 113458
rastemax@pc-dg-033-11:~$ Image trouvée
/users/Stockage-HDD/images-kvm/S2.03/i

rastemax@pc-dg-033-11:~$ ls /var/
backups/ cache/   lib/     local/   lo
rastemax@pc-dg-033-11:~$ telnet localh
Trying ::1...
Connection failed: Connection refused
Trying 127.0.0.1...
telnet: Unable to connect to remote ho
rastemax@pc-dg-033-11:~$ telnet localh
Trying ::1...
Connection failed: Connection refused
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Date: Fri, 03 May 2024 12:09:27 GMT
Server: Apache/2.4.57 (Debian)
Last-Modified: Mon, 08 Apr 2024 14:29:
ETag: "29cd-61596a2c57f24"
Accept-Ranges: bytes
Content-Length: 10701
Vary: Accept-Encoding
Connection: close
Content-Type: text/html

Connection closed by foreign host.
```

```
New Tab   Split View
~
~
~
~
~
~
~
log file: erver-rastemax sshd[1032]:
^C
root@server-rastemax:~# systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: enabled)
     Active: active (running) since Fri 2024-05-03 07:41:01 EDT; 1h 8min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 475 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 509 (sshd)
      Tasks: 1 (limit: 4645)
     Memory: 8.3M
        CPU: 163ms
     CGroup: /system.slice/ssh.service
             └─509 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

May 03 07:41:01 server-rastemax sshd[509]: Server listening on 0.0.0.0 port 22.
May 03 07:41:01 server-rastemax sshd[509]: Server listening on :: port 22.
May 03 07:41:01 server-rastemax systemd[1]: Started ssh.service - OpenBSD Secure Sh
May 03 07:44:23 server-rastemax sshd[610]: Accepted password for rastemax from 10.0
May 03 07:44:23 server-rastemax sshd[610]: pam_unix(sshd:session): session opened f
May 03 07:44:23 server-rastemax sshd[610]: pam_env(sshd:session): deprecated readin
```

From now on, and for more simplicity, you can use it on a shell directly from your real machine by typing the command **ssh the_username_you_choose@localhost -p 2222**.

II. Install Apache, PostgreSQL and PHP

Make sure you're already in a shell rooted

1. Install Apache

To get apache, we will enter "apt install apache2" and type yes when it asks you.

Once it's done, you can see if it's already running with **systemctl status apache2**.

If you don't have something like this:

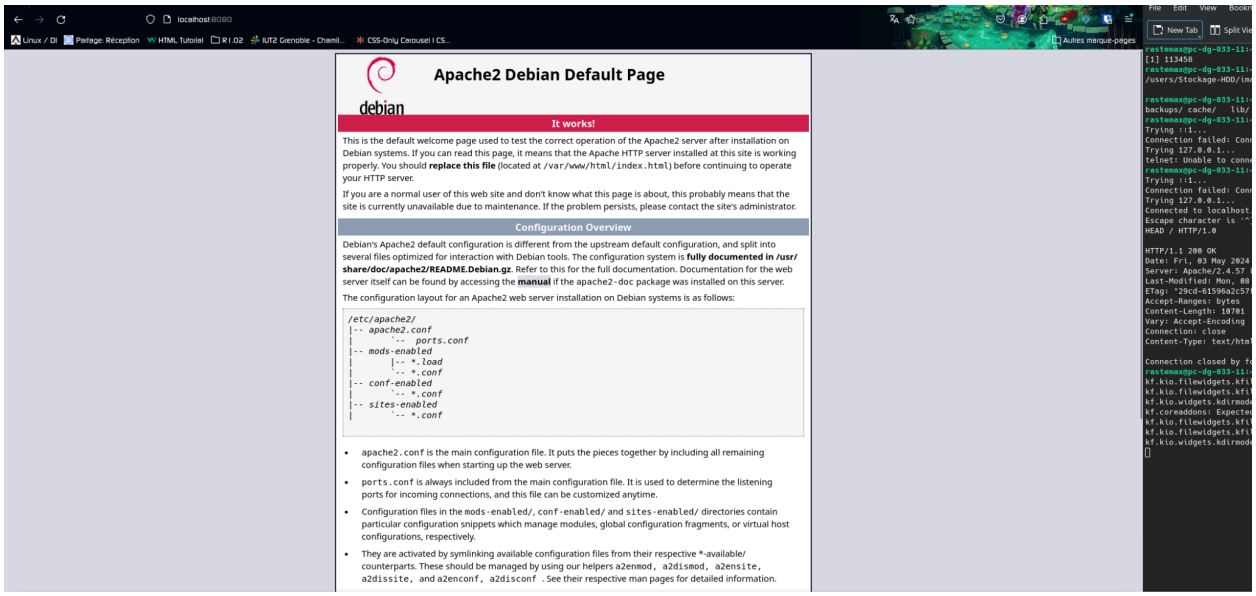
```
root@server-rastemax:~# systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
     Active: active (running) since Sun 2024-06-02 15:06:03 CEST; 1min 29s ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 971 (apache2)
      Tasks: 55 (limit: 3779)
     Memory: 13.2M
        CPU: 74ms
     CGroup: /system.slice/apache2.service
             ├─971 /usr/sbin/apache2 -k start
             ├─972 /usr/sbin/apache2 -k start
             └─974 /usr/sbin/apache2 -k start

Jun 02 15:06:03 server-rastemax systemd[1]: Starting The Apache HTTP Server...
Jun 02 15:06:03 server-rastemax apachectl[970]: AH00558: apache2: Could not reliably determine the >
Jun 02 15:06:03 server-rastemax systemd[1]: Started The Apache HTTP Server.
lines 1-16/16 (END)
```

You should type **systemctl start apache2**.

2. Accessing to Apache

To check if the server is open, you can type **telnet localhost 80**, and "**HEAD / HTTP/1.0**", to enter you need to type two times. with this it will respond to you "**HTTP/1.1 200 OK**".

Now you can access a default page of the server by simply typing in a web browser in the navigation bar "http://localhost:8080/", you will normally see this page:



3. Install PostgreSQL

You start by doing the same as Apache : **apt install postgres** and type "**y**" when it asks you to confirm.

To check that the service runs, you can type **systemctl status postgresql** and if it's not something like that :

```
root@server-rastemax:~# systemctl status postgresql@15-main.service
● postgresql@15-main.service - PostgreSQL Cluster 15-main
     Loaded: loaded (/lib/systemd/system/postgresql@.service; enabled-runtime; preset: enabled)
     Active: active (running) since Fri 2024-05-03 07:41:05 EDT; 1h 11min ago
    Process: 473 ExecStart=/usr/bin/pg_ctlcluster --skip-systemctl-redirect 15-main start (code=exited, status=0/SUCCESS)
   Main PID: 529 (postgres)
      Tasks: 6 (limit: 4645)
     Memory: 49.5M
        CPU: 2.679s
     CGroup: /system.slice/system-postgresql.slice/postgresql@15-main.service
             ├─529 /usr/lib/postgresql/15/bin/postgres -D /var/lib/postgresql/15/main -c config_file=/etc/postgresql/15/main/postgresql.conf
             ├─566 "postgres: 15/main: checkpointer "
             ├─567 "postgres: 15/main: background writer "
             ├─576 "postgres: 15/main: walwriter "
             ├─577 "postgres: 15/main: autovacuum launcher "
             └─578 "postgres: 15/main: logical replication launcher "

May 03 07:41:01 server-rastemax systemd[1]: Starting postgresql@15-main.service - PostgreSQL Cluster 15-main...
May 03 07:41:01 server-rastemax postgresql@15-main[473]: Removed stale pid file.
May 03 07:41:05 server-rastemax systemd[1]: Started postgresql@15-main.service - PostgreSQL Cluster 15-main.
root@server-rastemax:~#
```

Then you can type **systemctl start postgresql**.

4. Setting up PostgreSQL

Once you have done the steps above, you won't be able to connect to PostgreSQL with ssh, you will need to change some settings.

a. Giving access to PosgreSQL to someone else than the Virtual Machine

By default, we can only connect locally to PostgreSQL , which means that we can only access it through the machine that hosts the PostgreSQL server, so we need to change that in order to make it accessible by the users.

To make that possible we will edit two files.

First, we need to be in a shell root, once it's done, you can type

```
nano /etc/postgresql/15/main/postgresql.conf
```

Go down until you find these lines:

```
#listen_addresses = 'localhost'         # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost'; use '*' for all
                                        # (change requires restart)
port = 5432                             # (change requires restart)
max_connections = 100                   # (change requires restart)
#superuser_reserved_connections = 3     # (change requires restart)
unix_socket_directories = '/var/run/postgresql' # comma-separated list of directories
                                        # (change requires restart)
#unix_socket_group = ''                 # (change requires restart)
```

Uncomment the first line and change 'localhost' to '*' like this:

```
# - Connection Settings -

listen_addresses = '*'                  # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost'; use '*' for all
                                        # (change requires restart)
port = 5432                             # (change requires restart)
max_connections = 100                   # (change requires restart)
#superuser_reserved_connections = 3     # (change requires restart)
unix_socket_directories = '/var/run/postgresql' # comma-separated list of directories
                                        # (change requires restart)
#unix_socket_group = ''                 # (change requires restart)
```

Save and exit after that.

Now we need to allow only people we want with a password encrypted.

Type nano /etc/postgresql/15/main/pg_hba.conf and add the lines :

“#IPv4 remote connections:
host all all 0.0.0.0/0 scram-sha-256”
It should look like this:

```
# "local" is for Unix domain socket connections only
local   all             all                                     peer
# IPv4 local connections:
host    all             all             127.0.0.1/32            md5
# IPv6 local connections:
host    all             all             ::1/128                 md5
# Allow replication connections from localhost, by a user with the
# replication privilege.
local   replication     all                                     peer
host    replication     all             127.0.0.1/32            md5
host    replication     all             ::1/128                 md5
#IPv4 remote connections:
host    all             all             0.0.0.0/0               scram-sha-256
```

Now we need to restart the service to apply changes :

service postgresql restart

b. Creating a user

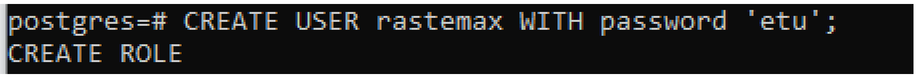
We’ve made it possible to log in with another computer, however, by default, there is only one user on PostgreSQL, it’s “postgres”, so we need to create one because this one is for the administrator only.

To create a new user, we need to access the user postgres by typing in a shell root **su - postgres** it allows you to access the user that can connect to PostgreSQL with administrator rights.

Once you’re here, you just type **psql** to access PostgreSQL. You should see something like that:

```
rastemax@server-rastemax:~$ psql base1
psql (15.6 (Debian 15.6-0+deb12u1))
Type "help" for help.

base1=>
```

Now we will type CREATE USER user_name WITH password ‘password’;



```
postgres=# CREATE USER rastemax WITH password 'etu';
CREATE ROLE
```

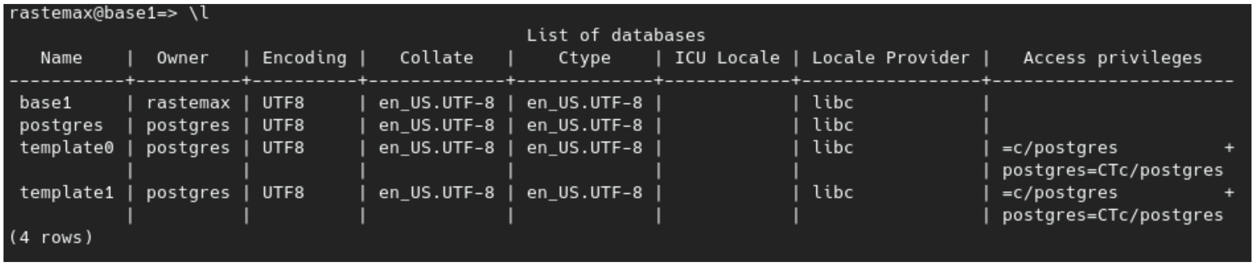
It's done, you have a new user which can connect to a database.

c. Creating a database and allowing the user to connect to it

You'll need to create a database where the user can connect and create tables.

Still in 'psql', we type `CREATE DATABASE db WITH owner=user_name`

You can see if the database is owned by the user by typing `\l`:



```
rastemax@base1=> \l
                                                 List of databases
   Name    |  Owner   | Encoding |   Collate   |    Ctype    | ICU Locale | Locale Provider |   Access privileges
-----------+----------+----------+-------------+-------------+------------+-----------------+-----------------------
 base1     | rastemax | UTF8     | en_US.UTF-8 | en_US.UTF-8 |            | libc            |
 postgres  | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |            | libc            |
 template0 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |            | libc            | =c/postgres          +
           |          |          |             |             |            |                 | postgres=CTc/postgres
 template1 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |            | libc            | =c/postgres          +
           |          |          |             |             |            |                 | postgres=CTc/postgres
(4 rows)
```

You can see many databases, they only appear for the user postgres. On top of that, we can see the database base1 I've created with the command above and it's the user rastemax that owns this database.

You can also check if the password is encrypted by typing `SELECT username, passwd FROM pg_shadow;` with the postgres user:

```
postgres=# SELECT usename,passwd FROM pg_shadow
;
 usename  |                                                             passwd
----------+---------------------------------------------------------------------------------------------------------------------------------------
 postgres |
 rastemax | SCRAM-SHA-256$4096:LwCGBjtcv4lFgHOqm67chA==$yNzxUwY9Ci3UF8v+UHli2+ohGZ0YgAYY7eX+LxesegE=:0ypOHLqZztayp9pKeHxLEHfBh9iUPL7bNy32S2wBDns=
(2 rows)
```

You can see the 2 users on the PostgreSQL server and the user "rastemax" with an encrypted password like we ask.

Now you can connect to the database with this command:

`psql -h localhost db -U user_name`

```
rastemax@pc-dg-033-11:~$ psql -h localhost base1
Password for user rastemax:
psql (15.6 (Debian 15.6-0+deb12u1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off)
Type "help" for help.

rastemax@base1=>
```

Now it's done, you can access PostgreSQL remotely and create tables.

To exit PostgreSQL you need to type \q.

To leave the postgres user on the Virtual Machine, you can just type exit.

5. Install PHP

To install PHP, we need to go in a shell root and type apt install php

To try if it has been installed successfully, you can create a file info.php with this code:

```
<?php

phpinfo();

phpinfo(INFO_MODULES);

?>
```

and you put it in /var/www/html so with the command :

nano /var/www/html/info.php (in a shell root) and see the result on http://localhost:8080/info.php, it should look like this:

PHP Version 7.4.33

System	Linux server-rastemax 5.10.0-29-amd64 #1 SMP Debian 5.10.216-1 (2024-05-03) x86_64
Build Date	Apr 12 2024 00:02:16
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/apache2
Loaded Configuration File	/etc/php/7.4/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/apache2/conf.d
Additional .ini files parsed	/etc/php/7.4/apache2/conf.d/10-opcache.ini, /etc/php/7.4/apache2/conf.d/10-pdo.ini, /etc/php/7.4/apache2/conf.d/20-calendar.ini, /etc/php/7.4/apache2/conf.d/20-ctype.ini, /etc/php/7.4/apache2/conf.d/20-exif.ini, /etc/php/7.4/apache2/conf.d/20-ffi.ini, /etc/php/7.4/apache2/conf.d/20-fileinfo.ini, /etc/php/7.4/apache2/conf.d/20-ftp.ini, /etc/php/7.4/apache2/conf.d/20-gettext.ini, /etc/php/7.4/apache2/conf.d/20-iconv.ini, /etc/php/7.4/apache2/conf.d/20-json.ini, /etc/php/7.4/apache2/conf.d/20-phar.ini, /etc/php/7.4/apache2/conf.d/20-posix.ini, /etc/php/7.4/apache2/conf.d/20-readline.ini, /etc/php/7.4/apache2/conf.d/20-shmop.ini, /etc/php/7.4/apache2/conf.d/20-sockets.ini, /etc/php/7.4/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.4/apache2/conf.d/20-sysvsem.ini, /etc/php/7.4/apache2/conf.d/20-sysvshm.ini, /etc/php/7.4/apache2/conf.d/20-tokenizer.ini
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902,NTS
PHP Extension Build	API20190902,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
DTrace Support	available, disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters	zlib.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, convert.iconv.*

This program makes use of the Zend Scripting Language Engine:
Zend Engine v3.4.0, Copyright (c) Zend Technologies
with Zend OPcache v7.4.33, Copyright (c), by Zend Technologies

Configuration

apache2handler

Now you can use apache with php and PostgreSQL server.

III. Installing PhpPgAdmin

Basically, you can only access PostgreSQL through a command line interface, but I will show you how you can use a web interface instead.

To do that we will install PhpPgAdmin in a shell root:

```
apt install phppgadmin
```

Now we need to set it up.

To make sure that it will support PostgreSQL 15, we will edit the file Connection.php :

nano /usr/share/phppgadmin/classes/database/Connection.php

Change the line

case '14': return 'Postgres';break;

to case '15': return 'Postgres';break;

```
switch (substr($version,0,2)) {
    case '15': return 'Postgres';break;
    case '13': return 'Postgres13';break;
    case '12': return 'Postgres12';break;
    case '11': return 'Postgres11';break;
    case '10': return 'Postgres10';break;
```

It will support version 15 but when you try to access it through the page http://localhost:8080/phppgadmin , it will say 403 Forbidden.

To fix that, we need to modify a file in /etc/apache/conf-available/phppgadmin.conf

In a shell root, type nano /etc/apache/conf-available/phppgadmin.conf

and change the line Require local

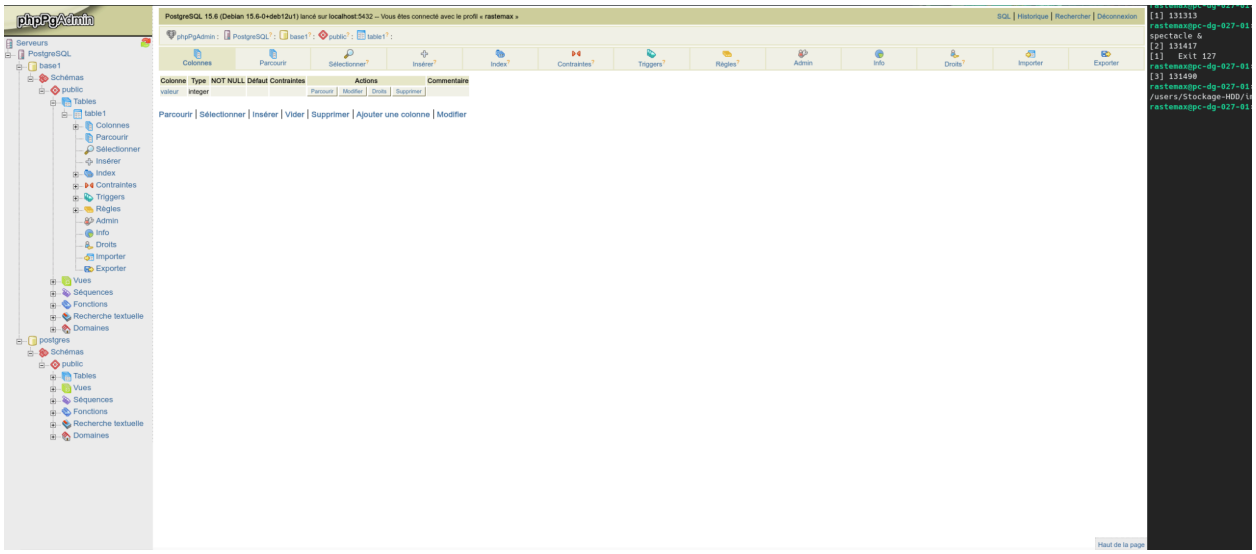
```
# Only allow connections from localhost:
Require local
```

to Require all granted.

```
# Only allow connections from localhost:
Require all granted
```

Once it's done, save(Ctrl + s) and close (Ctrl + x) and type systemctl restart apache2.service to make the changes effective.

Now you can access to PhpPgAdmin in your browser and connect with the user you've created earlier in to the database.



IV. Extra steps

1. Knowing the specifications

You may want to know further information about your server, you can do it by copying the file page_sae_S2.03.php into the directory /var/www/html on your server and opening it in http://localhost:8080/page_sae_S2.03.php.

```
     Tasks: 13 (limit: 4645)
    Memory: 38.1M
       CPU: 1.214s
    CGroup: /system.slice/apache2.service
            |- 531 /usr/sbin/apache2 -k start
            |- 534 /usr/sbin/apache2 -k start
            |- 537 /usr/sbin/apache2 -k start
            |- 540 /usr/sbin/apache2 -k start
            |- 614 /usr/sbin/apache2 -k start
            |- 637 /usr/sbin/apache2 -k start
            |- 638 /usr/sbin/apache2 -k start
            |- 653 /usr/sbin/apache2 -k start
            |- 654 /usr/sbin/apache2 -k start
            |- 655 /usr/sbin/apache2 -k start
            |- 656 /usr/sbin/apache2 -k start
            |-1000 sh -c "systemctl status apache2"
            `-1001 systemctl status apache2

My postgresql install is

ii  postgresql                  15+248                  all          object-relational SQL database (supported version)
ii  postgresql-15               15.6-0+deb12u1          amd64        The World's Most Advanced Open Source Relational Database
ii  postgresql-client-15        15.6-0+deb12u1          amd64        front-end programs for PostgreSQL 15
ii  postgresql-client-common    248                     all          manager for multiple PostgreSQL client versions
ii  postgresql-common           248                     all          PostgreSQL database-cluster manager

My postgresql status is

* postgresql.service - PostgreSQL RDBMS
     Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; preset: enabled)
     Active: active (exited) since Mon 2024-05-27 09:46:52 EDT; 51min ago
    Process: 592 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
   Main PID: 592 (code=exited, status=0/SUCCESS)
        CPU: 1ms

My ssh install is

ii  libssh2-1:amd64             1.10.0-3+b1             amd64        SSH2 client-side library
ii  openssh-client              1:9.2p1-2+deb12u2       amd64        secure shell (SSH) client, for secure access to remote machines
ii  openssh-server              1:9.2p1-2+deb12u2       amd64        secure shell (SSH) server, for secure access from remote machines
ii  openssh-sftp-server         1:9.2p1-2+deb12u2       amd64        secure shell (SSH) sftp server module, for SFTP access from remote machines
ii  task-ssh-server             3.73                    all          SSH server

My ssh status is

* ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: enabled)
     Active: active (running) since Mon 2024-05-27 09:46:49 EDT; 51min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 489 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 510 (sshd)
      Tasks: 1 (limit: 4645)
     Memory: 8.2M
        CPU: 215ms
    CGroup: /system.slice/ssh.service
            `-510 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"
```

```
     Tasks: 13 (limit: 4645)
     Memory: 38.1M
        CPU: 1.214s
     CGroup: /system.slice/apache2.service
             |- 531 /usr/sbin/apache2 -k start
             |- 534 /usr/sbin/apache2 -k start
             |- 537 /usr/sbin/apache2 -k start
             |- 540 /usr/sbin/apache2 -k start
             |- 614 /usr/sbin/apache2 -k start
             |- 637 /usr/sbin/apache2 -k start
             |- 638 /usr/sbin/apache2 -k start
             |- 653 /usr/sbin/apache2 -k start
             |- 654 /usr/sbin/apache2 -k start
             |- 655 /usr/sbin/apache2 -k start
             |- 656 /usr/sbin/apache2 -k start
             |-1000 sh -c "systemctl status apache2"
             `-1001 systemctl status apache2
```

My postgresql install is

```
ii  postgresql                  15+248                      all          object-relational SQL database (supported version)
ii  postgresql-15               15.6-0+deb12u1              amd64        The World's Most Advanced Open Source Relational Database
ii  postgresql-client-15        15.6-0+deb12u1              amd64        front-end programs for PostgreSQL 15
ii  postgresql-client-common    248                         all          manager for multiple PostgreSQL client versions
ii  postgresql-common           248                         all          PostgreSQL database-cluster manager
```

My postgresql status is

```
* postgresql.service - PostgreSQL RDBMS
     Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; preset: enabled)
     Active: active (exited) since Mon 2024-05-27 09:46:52 EDT; 51min ago
    Process: 592 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
   Main PID: 592 (code=exited, status=0/SUCCESS)
        CPU: 1ms
```

My ssh install is

```
ii  libssh2-1:amd64             1.10.0-3+b1                 amd64        SSH2 client-side library
ii  openssh-client              1:9.2p1-2+deb12u2           amd64        secure shell (SSH) client, for secure access to remote machines
ii  openssh-server              1:9.2p1-2+deb12u2           amd64        secure shell (SSH) server, for secure access from remote machines
ii  openssh-sftp-server         1:9.2p1-2+deb12u2           amd64        secure shell (SSH) sftp server module, for SFTP access from remote machines
ii  task-ssh-server             3.73                        all          SSH server
```

My ssh status is

```
* ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: enabled)
     Active: active (running) since Mon 2024-05-27 09:46:49 EDT; 51min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 489 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 510 (sshd)
     Tasks: 1 (limit: 4645)
     Memory: 8.2M
        CPU: 215ms
     CGroup: /system.slice/ssh.service
             `-510 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"
```

You can see on this page the disk partitions and check with /sbin/blkid if they are correct :

```
root@server-rastemax:~# /sbin/blkid
/dev/sda5: UUID="f39349d4-bc78-401d-beff-d02408a88140" TYPE="swap" PARTUUID="d37ca061-05"
/dev/sda1: UUID="a3ef6f9e-3bf5-4ac1-b82a-c0a436845f88" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="d37ca061-01"
```

You can also see the network interface from your server and the specification and the version of the Apache server, PostgreSQL server and ssh server.

2. Checking the available storage

Once you've done everything in this Handbook, you can check if the available storage looks like this by typing df -h:

```
rastemax@server-rastemax:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            1.9G     0  1.9G   0% /dev
tmpfs           392M  484K  392M   1% /run
/dev/sda1       3.0G  1.6G  1.3G  56% /
tmpfs           2.0G  1.1M  2.0G   1% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           392M     0  392M   0% /run/user/1000
```

3. Keep your server secure

To keep your server the most secure, you should regularly update the packages to reduce breach in your system, you can do it easily by typing in a shell root apt update which allows to update the packages list and apt upgrade to install the update.

Conclusion

Now you have a fully operational server working with Debian 12 and made for hosting web servers made in php, PostgreSQL server and you know how you can manage it and keep it the most secure.